ENQPY™ · PLAIN-LANGUAGE CLAIM STATEMENT

What Enqpy™ proves, in plain terms.
Companion to the Enqpy™ proof materials · NQP LLC · Rev 5.0 · 2026

The claim, in plain terms

Enqpy™ is a high-performance stream cipher with a formally proved core property: from the ciphertext alone, an adversary is left with a vast field of equally consistent plaintexts and cannot pick out the right one. That uncertainty does not come from assuming any computation is hard — so no amount of computation, quantum included, erodes it. Capture the ciphertext and keep it forever: on its own, it never resolves to the message.
Run it the way it ships — each record under its own fresh key, authenticated — and that property is what your deployment carries.

What the proof establishes

Two results, both from captured ciphertext alone, plus a real implementation:

- **The message stays uncertain (the primary result).** At the HIGH profile (n = 64), the set of plaintexts consistent with a given ciphertext is at least 2^{128}, and uniform — so the min-entropy of the message given the ciphertext is at least 128 bits. The true plaintext sits in a field of 2^{128} equally good ones, with nothing in the ciphertext to single it out.
- **The key stays uncertain.** A fixed set of four master keys produces identical ciphertext for every plaintext, nonce, and session, so from the ciphertext alone the key never resolves below those four — a $\log_2 4 = 2$-bit equivocation floor that no transcript, of any length, erodes. Pinning it to exactly those four (exactly two bits, no more) is the tightness that holds when the adversary also has the matching plaintext; from ciphertext alone the floor can only be larger.
- **No hardness assumption.** These bounds are algebraic. They do not rest on any problem being hard to solve, so more computation does not touch them — and because Enqpy™ is symmetric, the public-key assumptions quantum algorithms break (factoring, discrete log) are not in the picture. Bring the quantum computer; it does not help.
- **A real cipher.** The core is a concrete reference implementation — single-file C11, no dependencies — fast in portable C, faster in hardware, passing the full NIST SP 800-22 statistical suite. Throughput figures are in the benchmark report.

What kind of secrecy this is

Shannon, in 1949, separated two properties. Perfect Secrecy — the one-time pad — needs a key as long as the message; Enqpy™ does not claim it. The second property is equivocation: given any amount of ciphertext, the adversary is left with many plausible plaintexts and cannot identify one. Enqpy™ proves the second, with a finite, reusable key. That is the property cryptography has wanted to make practical since 1949, and it is what is on the page in the proof.

How you run it

Each record gets its own fresh, independent key, with authentication (encrypt-then-MAC); a long message is simply several records. A known-plaintext exposure in one record stays in that record — it has no path into the next. This follows directly from the proved ciphertext-only results applied per record; with independently keyed records it adds no new assumption, and it is the same per-record hygiene every modern cipher deploys under.
The proof also maps the boundary precisely, so you can see it for yourself: under one key reused across fully-known plaintext, exposure begins to transfer at about two fully-known 2,048-byte

windows within a record. That is the line that each-record-its-own-key keeps you on the right side of — stated here because we mapped it on purpose, and because you should be able to find it in the proof and confirm it.

Where to verify

- **The proof.** The formal argument for the message-axis and key-axis equivocation, with the derivations and appendices.
- **The toy run (R = 2, n = 4).** Every (EK, QK) pair enumerated by hand at small size, so you can watch, in actual numbers, the ciphertext fail to single out a plaintext. The proof generalizes it; the toy run shows it arithmetically.
- **The reference implementation.** Single-file C11, no external dependencies, with self-test assertions against the published vectors. Portable from 8-bit microcontrollers to 64-bit servers.
- **The covenant.** NQP LLC owns the Enqpy™ patents and has published a dated, irrevocable, successor-binding covenant not to assert them against use of the cipher as defined in the covenant — study, research, evaluation, open implementation, standards-track adoption, and commercial deployment at any scale. The pledge is itself a public, verifiable artifact.

Run the code. Read the proof. Find the line yourself.

Inquiries: RPM@enqpy.com · Public release: June 1, 2026
Materials: enqpy.com, github.com/nqp-llc/enqpy.